

Note: Print your answers clearly. Show processes to get partial credits.

1. (i) Convert the binary to hex:

$$111\ 1011 = \underline{7B}$$

(ii) Convert the hex to binary:

$$102 = \underline{0001\ 0000\ 0010}$$

2. (i) Convert the *unsigned* hex number 82_h to decimal number.

$$82_h = 1000\ 0010$$

$$2^7 + 2^1 = 128 + 2 = 130$$

(ii) Convert the *signed* hex number 82_h to decimal number.

$$82_h = 1000\ 0010$$

Since MSB = 1, so it is negative. Find its absolute value with two's complement method:

$$\begin{aligned} 0111\ 1101 + 1 &= 0111\ 1110 \\ &= 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 = 64 + 32 + 16 + 8 + 4 + 2 = 126 \end{aligned}$$

$$\text{Thus, } 82_h = -126$$

3. (i) Convert the decimal -1 to 8-bit binary.

$$1 = 0000\ 0001$$

$$\text{Find its two's complement } -1 = 1111\ 1110 + 1 = 1111\ 1111$$

$$\text{Thus, } -1 = 1111\ 1111$$

(ii) Convert the decimal -12 to 8-bit binary.

$$12 = 8 + 4 = 0000\ 1000 + 0000\ 0100 = 0000\ 1100$$

$$\text{Find its two's complement } -12 = 1111\ 0011 + 1 = 1111\ 0100$$

$$\text{Thus, } -12 = 1111\ 0100$$

4. Find the results of the flags (OV/NV, PL/NG, ZR/NZ, CY/NC) and AL after the executions of the code segments in MS-DOS Debug.

(i)

```
MOV AL, 7E
ADD AL, 2
```

$$7E + 2 = 0111\ 1110 \text{ (positive)} + 0000\ 0010 = 1000\ 0000 \text{ (negative)}$$

For a 8-bit binary, its valid range is: -128 (1000 0000=80_h) to 127 (0111 1111=7F_h). 7E + 2 will be over the max. value of the range. Therefore, overflow will happen.

a) OV/NV OV b) PL/NG NG c) ZR/NZ NZ d) CY/NC NC e) AL 80

(ii)

```
MOV AL, 62
SUB AL, 9
```

$$62 - 9 = (60+2) - 9 = 60 - 7 = (5F + 1) - 7 = 5F - 6 = (5E+1) - 6 = 5E - 5 = \dots = 59 \text{ (positive)}$$

Since the result 59 is in the range (80-7F), no overflow.

a) OV/NV NV b) PL/NG PL c) ZR/NZ NZ d) CY/NC NC e) AL 59

5. (i) The instruction execution cycle includes some or all of the five operations (in order): 1) fetch; 2) decode; 3) fetch operands; 4) execute; 5) store output. Indicate the operations for executing the following instructions:

a) ADD DX, [100]

Every instruction execution needs 1), 2) and 4). This one needs 3) fetch operand stored at RAM location [100]. The result is in DX that is inside CPU, so no 5)

Operations: 1, 2, 3, and 4

b) INC CX

Operations: 1, 2, and 4

(ii) Convert the logic addresses to absolute addresses of memory: 08FF:012A

0	8	F	F	0
	0	1	2	A
0	9	1	1	A

The absolute address is 0911A

6. (i) What is the size in KB or MB of ROM whose linear address is from 00000 to FFFFF. Show the calculation process.

$$1 + (FFFFFF - 00000) = 10\ 0000_h = 1\ 0000\ 0000\ 0000\ 0000$$

$$= 2^{20}$$

So the memory access is 1MB.

(ii) What is the size in KB or MB of ROM whose linear address is from C0000 to FFFFF. Show the calculation process.

$$1 + (FFFFFF - C0000) = 1 + 3FFFF = 40000 = 0100\ 0000\ 0000\ 0000\ 0000$$

$$= 2^{18} = 2^8 * 2^{10} = 256 * 2^{10}$$

So the memory access is 256 KB.

7. (i) What are the values of the registers after the following assembly code is executed with MS-DOS Debug?

```

MOV AX, 20 ;    AX: 20
MOV BX, 6  ;    BX: 6
MOV CX, 35 ;    CX: 35
PUSH AX   ;    Stack (top → bottom): 20
PUSH CX   ;    Stack (top → bottom): 35, 20
PUSH BX   ;    Stack (top → bottom): 6, 35, 20

POP AX    ;    AX: 6
POP CX    ;    CX: 35
POP BX    ;    BX: 20

```

AX: 6 BX: 20 CX: 35

(ii) What are the values of register DL and the memory address [203] after the following operations with MS-DOS Debug. Show the calculation process.

```
-A 200
DB 99, 22, 33, 00
```

RAM allocation -

```
[200]: 99
[201]: 22
[202]: 33
[203]: 00
```

```
-A 100
MOV DL, [200]      ; DL: 99
SUB DL, [201]      ; 99 - 22 = 77 is stored in DL
ADD DL, [202]      ; 77 + 33 = AA is stored in DL
MOV [203], DL      ; [203]: AA
INC DL             ; DL: AA + 1 = AB
```

DL: AB DS:[203] AA

8. What are the values of the registers after the following assembly code is executed with MS-DOS Debug? Fill “U” if the value is unknown.

```
MOV AX, 9
MOV BX, 6
MOV CX, 0
MOV DX, 0
DIV BX
```

DX:AX is divided by BX – quotient is stored in AX and the remainder is stored in DX

$$9 / 6 = 1 \frac{3}{6}$$

The quotient is 1 (AX) and the remainder is 3 (DX). Values in other registers are not changed.

AX: 1 BX: 6 CX: 0 DX: 3